Vario-schaal gegevens in een Geoweb context

Huang, Lina; Meijers, Martijn; Suba, R.; van Oosterom, P.J.M.

Publication date
2017

Document Version
Final published version

Published in
Geo-Info

Vario-schaal gegevens in

Zo'n vijf jaar gelden is in Geo-Info het concept van vario-schaal geo-informatie beschreven (Van Oosterom en Meijers, 2012). In dit eerdere artikel werd de eerste echt geleidelijke vario-schaal structuur gepresenteerd: een delta schaal geeft een delta in de kaart (en hoe kleiner de delta schaal hoe kleiner de delta kaart). De afgelopen vijf jaar is er veel R&D verricht om het concept van vario-schaal geo-informatie te realiseren: ontwikkelen van prototypen en testen met echte data. In het kader van het Open Technologieprogramma (OTP van STW, Stichting Technische Wetenschappen) project 11185 'Vario-scale geo-information' is er de afgelopen jaren veel vooruitgang geboekt (zie bronnen). De belangrijkste resultaten zullen in een serie beknopte artikelen worden behandeld. Dit is het tweede artikel in de serie.

Door Lina Huang, Martijn Meijers, Radan Šuba en Peter van Oosterom

Vector data efficiënt overdragen over het web blijft een uitdaging, zeker als het gaat om een wijd bereik van kaartschalen: van zeer gedetailleerd (grootschalig) tot overzichtskaart (kleinschalig). In vergelijking met raster data, waar geavanceerde technieken voor multi-schaal gebruik voor handen zijn (denk aan: raster data piramide, wavelet compressie), levert vector data overdracht met meerdere kaartschalen nog de nodige uitdagingen op.

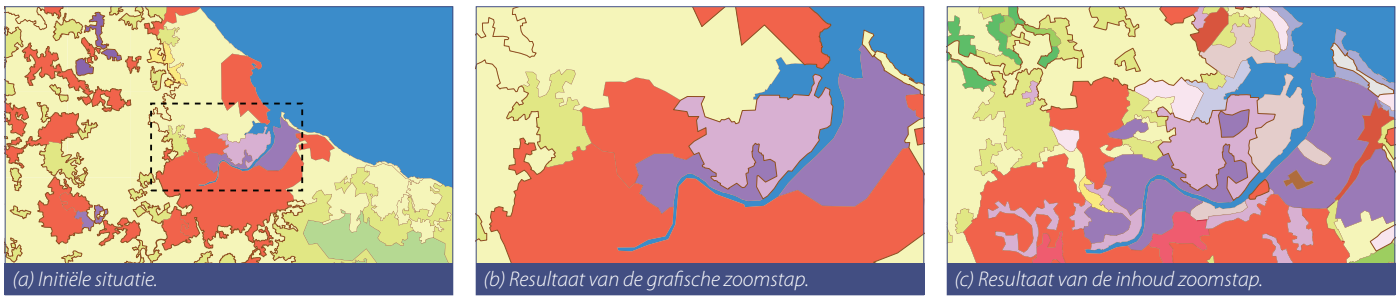
In potentie kun je met vector data toepassingen realiseren die interactiever zijn (denk aan: eenvoudig toepassen van eigen styling, transformatie naar ander coördinaatstelsel, selectie van en interactie met een of meer objecten, enzovoorts). Wanneer echter te veel en te gedetailleerde data overgestuurd moet worden, is er grote kans op vertragingen bij de verwerking van de data, zeker als dit over een netwerk moet waar slechts beperkte bandbreedte beschikbaar is. Daarnaast lopen eindgebruikers het risico hun geografische context kwijt te raken als een kaartapplicatie bij zoomacties grote stappen neemt (waarbij de kaart in één keer veel verandert). Een zoomactie waarbij de inhoud van het kaartbeeld stapsgewijs aangepast wordt, kan hierbij helpen, zeker in het geval van mobiel kaartgebruik (kleine schermpjes). Ook gebruikers die de kaartapplicatie op hun desktop-omgeving gebruiken, hebben baat bij deze graduele zoom-mogelijkheden, waarbij de kaart al tijdens een zoom-actie verandert naar de nieuwe gevraagde uitsnede met nieuwe inhoud.

Al met al is er dus de wens van flexibele web-overdracht van vector data, waarop traploos in- en uitgezoomd kan worden. Deze context is het uitgangspunt van ons onderzoek, dat probeert geleidelijke overdracht van vario-schaal data te realiseren door gebruik te maken van webservices. Het idee is om een vario-schaal datastructuur aan de server-kant in te richten, waarbij de inhoud van deze structuur het resultaat van een generalisatieproces is. In dit proces wordt het aantal vlakobjecten dat in een kaart aanwezig is stapje voor stapje minder. Ook worden de grenzen tussen de vlakken versimpeld.

We stellen in dit artikel drie verschillende architecturen voor om deze vario-schaal structuur te gebruiken in een web-context. Bij de laatste optie wordt vector data op een incrementele wijze van server naar client gestuurd. Dit leidt tot snellere visuele feedback bij de eindgebruiker: je kunt de kaart al zien veranderen, terwijl nog niet het hele antwoord binnen is (progressieve dataoverdracht).

Geleidelijk en traploos zoomen

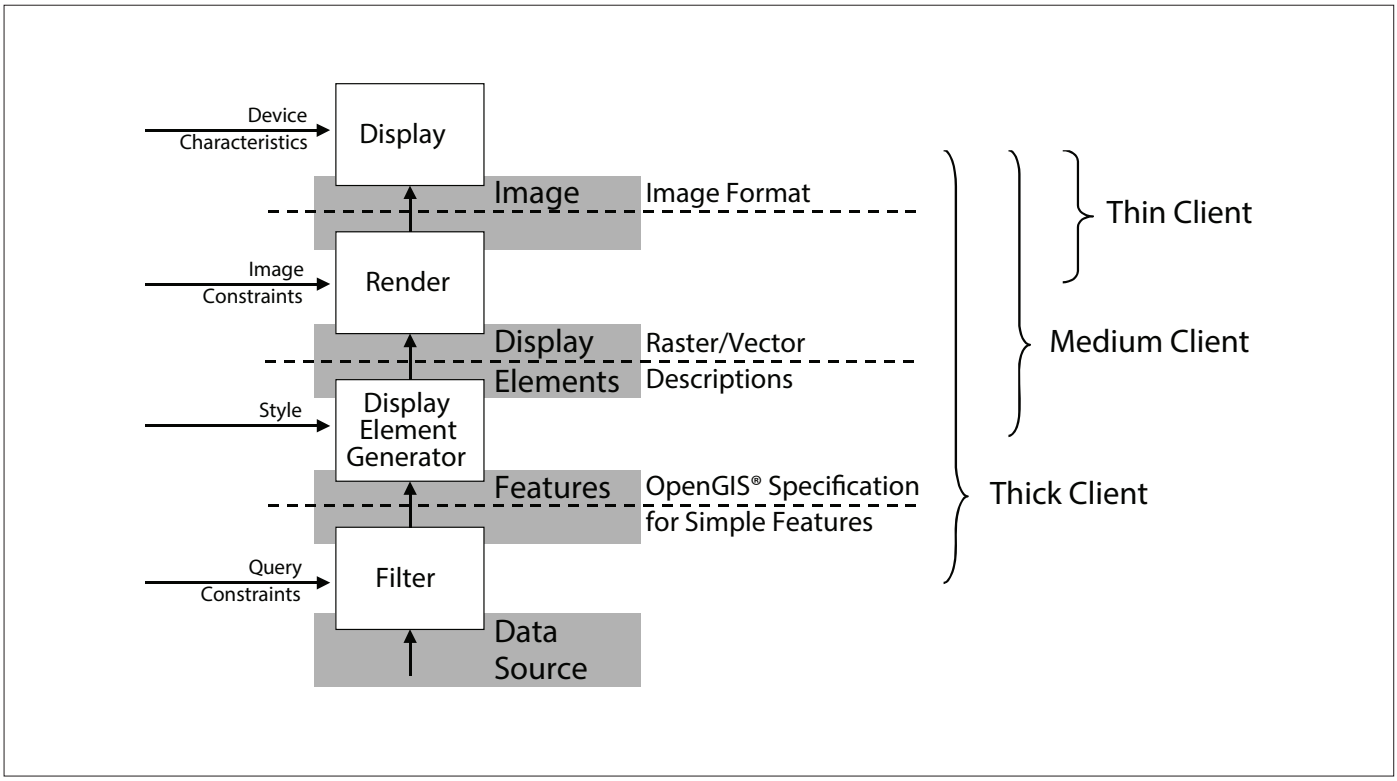
Om beter te kunnen definiëren wat geleidelijk zoomen inhoudt, analyseren we dit eerst in meer detail (figuur 1). Een gebruiker initieert een zoomactie en dit resulteert in een aantal stappen die door een kaartapplicatie genomen worden: de huidige uitsnede van de kaartinhoud wordt geschaald en eventueel verplaatst (op de al aanwezige vector objecten wordt schaling en translatie toegepast). Dit resulteert erin dat de oude kaartuitsnede na deze stap vergroot (of verkleind) is. De nieuwe kaartinhoud die nu te zien moet zijn, wordt ook direct aangevraagd bij de webservice. Nadat voor deze uitsnede data opgehaald is, wordt dit op het scherm gezet en verfijnt zo het oude kaartbeeld. De eerste stap die gezet wordt, is een grafische transformatie, terwijl de tweede stap de inhoud van het kaartbeeld vervangt. Binnen de zoomactie onderscheiden we dus a) een grafische zoom-stap en b) een inhoud zoom-stap. Door deze twee stappen af te wisselen en te zorgen dat de inhoud zoom-stap geen enorme veranderingen teweegbrengt, bereiken we het effect van geleidelijk zoomen. Merk op dat een zoomactie uit een serie van meerdere kleine grafische en inhoud zoom-stappen kan bestaan (waarbij grafische en inhoud zoom worden afgewisseld).



(a) Initiële situatie. (b) Resultaat van de grafische zoomstap. (c) Resultaat van de inhoud zoomstap.

Figuur 1 - Bij geleidelijk en traploos zoomen onderscheiden we 2 stappen: (b) een grafische zoomstap en (c) inhoud zoomstap.

een Geoweb context



Figuur 2 - Classificatie van verschillende clients (OGC, 2000).

De tGAP structuur zorgt voor het opslaan van het resultaat van stapsgewijze generalisatie operaties. Door deze operaties wordt de kaartinhoud simpeler en simpeler. Bij uitzoomen kan de beschrijving van het generalisatieproces in deze vorm direct gebruikt worden. Echter, bij inzoomen wordt de beschrijving omgedraaid en leidt zo tot toevoegen van detail aan de kaart. Deze inverse functie noemen we in het Engels: refinement. Kaart verfijnacties (refinement) leveren dus het toevoegen van detail aan de bestaande kaart op.

Thin of thick client?

Bij het gebruiken van kaarten in een geoweb-context, kunnen er vier stappen worden onderscheiden, voordat een kaartbeeld aan de kant van de client getoond kan worden (zie figuur 2):

1. selectie van de gegevens die getoond moeten worden,
2. gegevens omzetten naar grafische primitieven die gevisualiseerd kunnen worden,
3. transformeren van de grafische primitieven naar een rasterbeeld wat op het scherm gezet wordt en
4. het rasterbeeld daadwerkelijk op het scherm zetten.

De volgorde van deze stappen is vast. Stap 1 vindt altijd plaats op de server en stap 4 altijd op de client. Voor de andere stappen kan nog gekozen worden waar de stap wordt uitgevoerd. Bij een zogenaamde 'thin' client worden stappen 1 tot en met 3 op de server uitgevoerd, en alleen stap 4 op de client (aangezien voor het op het scherm zetten niet hele krachtige hardware benodigd is), terwijl een 'thick' client de stappen 2 tot en met 4 in zijn geheel voor zijn rekening neemt.

Aangezien bijvoorbeeld mobiele telefoons al heel krachtige rekenhardware aan boord hebben, zetten we in op een 'thick' client. Door topologische primitieven aan de serverkant uit de tGAP structuur te selecteren en deze op te sturen kan de overdracht over het netwerk efficiënt worden gerealiseerd (grenzen tussen vlakobjecten worden maar een keer overgestuurd). Echter, voordat vlakken getoond kunnen worden, moet een client dan wel deze vlakken uit de topologische structuur reconstrueren en omzetten naar grafische primitieven. Dit is geen probleem, omdat de client toch over de benodigde rekenkracht beschikt.

Gegeven de tGAP structuur met een goede vulling (zie het eerste artikel in deze serie in het vorige nummer van Geo-Info) en de wens om te werken met een 'thick' client rest dan de vraag: wat is een geschikte communicatie-architectuur, voor het efficiënt overdragen van de vario-schaal gegevens van server naar client? We hebben in een 'thick' client-setting drie verschillende opties onderzocht om vario-schaal data in een web-service setting te gebruiken.

Vraag-antwoord spel

In een webservice architectuur verloopt de communicatie als een vraag-antwoord spel. De client stuurt een verzoek (vraag) en hierop levert de server vervolgens een antwoord. We hebben drie opties (A, B en C) ontworpen, geïmplementeerd, getest en vervolgens met elkaar vergeleken.

Bij optie A stelt de client de vraag: voor deze uitsnede (2D bounding box) en deze schaal wil ik de juiste set aan topologische primitieven ontvangen om een kaart te kunnen maken. Het antwoord wat de server vervolgens geeft: hier heb je een set aan topologische primitieven, dit moet voldoende zijn om het gebied

te vullen met vlakken (die tevens het juiste detailniveau bevatten voor deze schaal).

Voor optie B is de vraag: voor een nieuwe uitsnede en schaal wil ik de benodigde topologische primitieven ontvangen om een nieuwe kaart te kunnen maken, maar ik wil niet opnieuw de primitieven ontvangen die ik al eerder ontving voor laatste ***n*** uitsnede-en-schaal-paren: (gebied$_1$, schaal$_1$), (gebied$_2$, schaal$_2$), enzovoorts. Antwoord: hier heb je een set aan topologische primitieven, dit moet voldoende zijn om de kaartuitsnede te vullen met vlakken samen met de topologische primitieven had voor (gebied$_1$, schaal$_1$), (gebied$_2$, schaal$_2$), enzovoorts.

En bij optie C is de vraag die de client stelt: voor een overgang tussen deze start uitsnede en start schaal wil ik de juiste topologische primitieven gesorteerd in goede volgorde ontvangen om de kaartinhoud, die ik al had, objectsgewijs te kunnen updaten en uit te komen op eind uitsnede en eind schaal. Antwoord: met deze stroom aan pakketjes van topologische primitieven kun je de kaartinhoud in stapjes updaten, zodat je uiteindelijk op de gewenste uitsnede en schaal uitkomt.

Om de vragen efficiënt te kunnen beantwoorden wordt de vraag van de client door de server omgezet in een database query. Binnen de database zijn de topologische primitieven geïndexeerd met een 3D R-tree (2D ruimte met 1D schaal) waardoor de database in staat is snel een antwoord te geven.

De opties in meer detail

Optie A is relatief simpel en rechttoe rechtaan: De inhoud van de kaart is precies afgestemd op wat de gebruiker wil zien, overeenkomstig hoe een WMS (Web Map Service) of WFS (Web Feature Service) vraag-antwoord verzoek werkt (voor respectievelijk een raster- of vectorkaart). Een client is hierbij overigens niet op de hoogte dat er een tGAP structuur op de server aanwezig is. Na elk verzoek wordt de hele opgebouwde topologische structuur vervangen. Optie B lijkt erg op optie A, maar met aanwezigheid van een cache aan de clientzijde (tijdelijke opslagstructuur). Aangezien een huidig verzoek van een gebruiker waarschijnlijk gerelateerd zal zijn aan een eerder verzoek (een gebruiker heeft de kaart bijvoorbeeld slechts een klein stukje verschoven/ingezoomd op de vorige situatie) probeert deze optie de vario-schaal gegevens die in het eerdere verzoek al zijn overgedragen (deels) te hergebruiken. Hiervoor is het nodig dat de client zich wel bewust is dat voor elke topologische primitieve een schaalbereik is gespecificeerd en de client dus de juiste primitieven uit eerdere antwoorden snel kan selecteren. Zowel optie A als optie B vervangen het hele kaartbeeld, alle objecten die in de kaart aanwezig zijn, in één keer na een zoom actie (dus niet geleidelijk).

Optie C: nadat een gebruiker een grafische zoom-stap heeft uitgevoerd, wordt een serie van instructies aangevraagd om progressief de inhoud van de kaart op object niveau aan te passen: de inhoud zoom-stappen, met nieuwe informatie, toe te voegen aan de bestaande kaart. Doordat de hele tGAP structuur bij deze oplossing ook bekendgemaakt wordt aan de client (en dus welke topologische primitieven gebruikt moeten worden) kan een client zelf de primitieven uit de kaart halen die voor een schaal niet meer benodigd zijn en de primitieven uit het antwoord toevoegen aan de huidige kaart. Optie C werkt het kaartbeeld op objectniveau bij en levert dus de meest geleidelijke van de drie voorgestelde oplossingen.

Voor de verschillende opties zijn de structuren die aan de clientkant nodig zijn enigszins verschillend. Voor optie A is alleen een topologische structuur nodig, op basis waarvan de client de vlakken kan vormen. Voor optie B is naast de topologische structuur ook een structuur nodig (een cache) waarmee bijgehouden wordt welke vraag eerder gesteld werd, en welk antwoord daarbij hoorde. Voor optie C is de meest geavanceerde structuur nodig, omdat topologische primitieven incrementeel uit de structuur gehaald/bijgeplaatst moeten kunnen worden, zodra ze (niet) meer benodigd zijn.

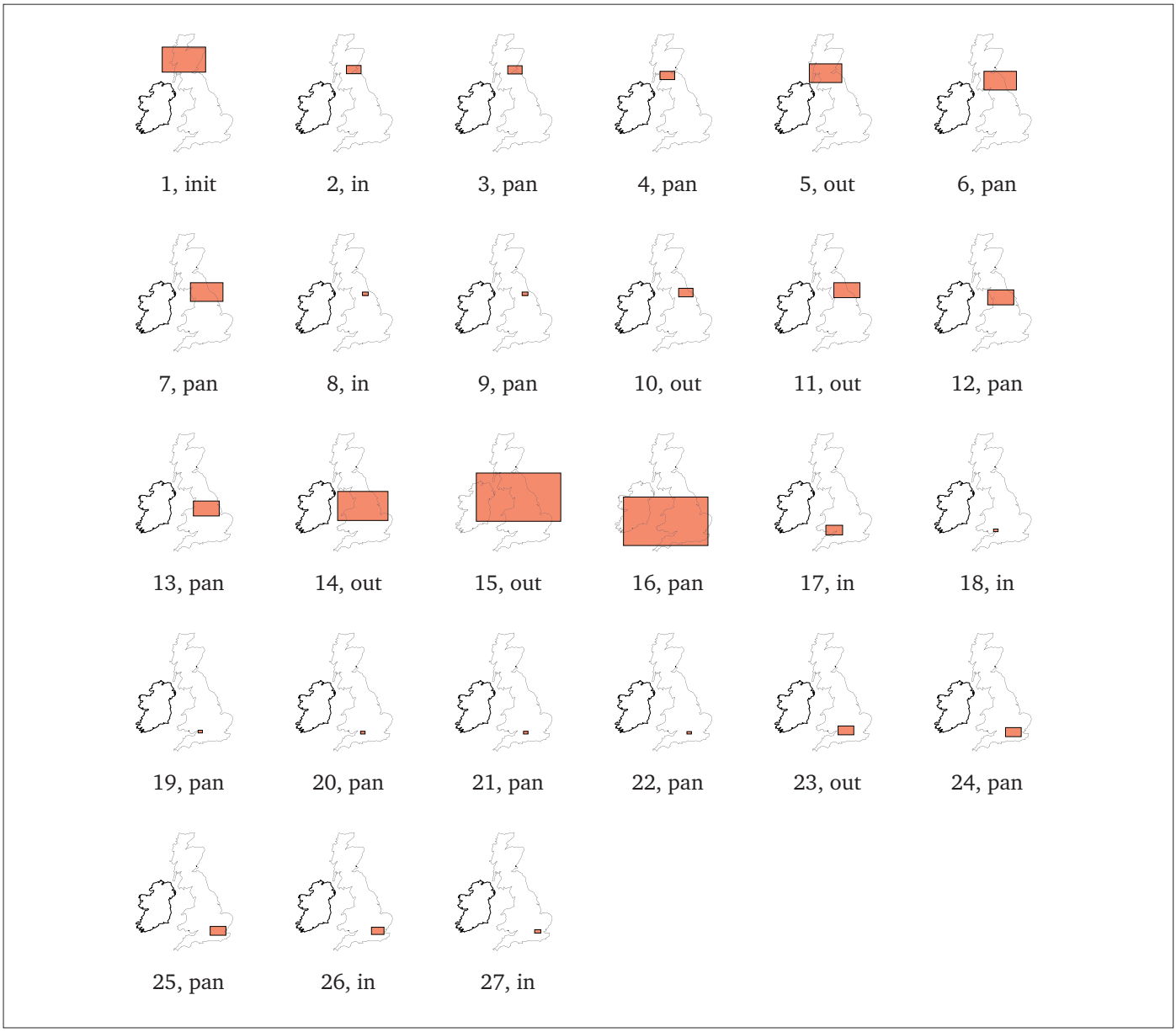
Benchmark experiment met landgebruiksdata

We hebben een deel van de Europese landgebruik dataset (Corine 2006) omgezet naar topologische primitieven en hier een tGAP structuur voor gebouwd. Vervolgens is getest met een interactiepad dat de gebruiker heeft afgelegd (figuur 3). De acties die voorkomen zijn: schuiven (pannen), inzoomen en uitzoomen, waarbij soms grote schaalstappen werden genomen en soms kleine. Gebruikersacties resulteren in een set van vragen, die afgevuurd kunnen worden op de vario-schaal server. Dit vraag-antwoord spel werd uitgevoerd door een automatisch proces, dat steeds twintig keer is uitgevoerd voor de verschillende opties.
Tijdens het experiment verzamelden we gegevens over de hoeveelheid overgedragen gegevens en de tijd die het bevragen van de server nam. De metingen voor de verwerkingstijd van het vraag-antwoord verzoek werden vervolgens geaggregeerd: gemiddelde, snelste en langzaamste verzoek. Bij de verwerkingstijd werden twee tijdstippen gemeten: de tijd die nodig is tot aan de start van de ontvangst van het antwoord (time to first byte) en de tijd die nodig is om het antwoord in zijn geheel te ontvangen (time to last byte, getoond in figuur 4(a)). Voor optie B zijn 4 varianten getest: B_0, B_1, B_2, en B_3, waarbij het nummer het aantal vorige verzoeken voor kaartuitsneden aangeeft. Qua hoeveelheid gegevens ligt gegevensgebruik voor de drie opties in nagenoeg dezelfde orde grootte (optie B_0 is iets duurder, omdat deze optie geen besparing kan realiseren door eerdere verzoeken, maar er worden wel stukken van de tGAP face boom overgestuurd, zoals nodig bij opties B en C, maar niet bij optie A). Qua verwerkingstijd functioneert optie C het best, optie A is tweede en optie B (met varianten 0-3) laatste. Het experiment laat zien dat elke vraag (ongeacht welke optie gebruikt is) beantwoord kan worden binnen twee seconden.

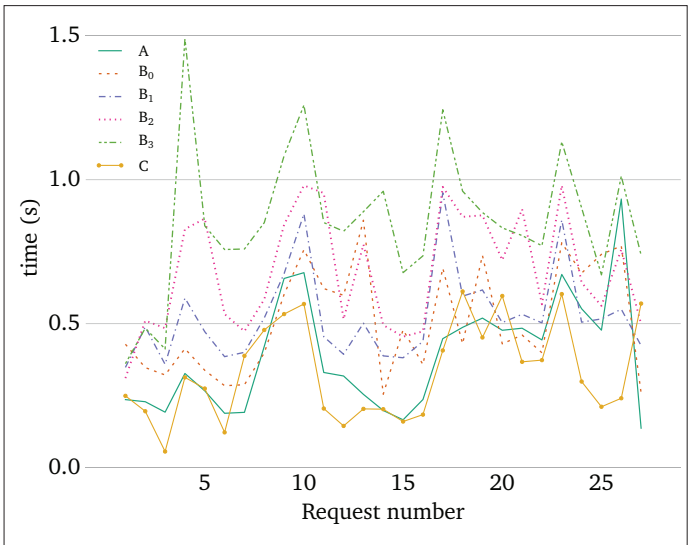
Optie A zet de basis, qua hoeveelheid dataoverdracht als vergelijk voor de andere opties. Nagenoeg dubbele overdracht van gegevens vindt bij deze optie plaats als de gebruiker de kaart slechts een klein beetje schuift of in/uitzoomt.

Voor B wordt aangegeven hoeveel eerdere kaartuitsneden door het proces werden onthouden (tot drie eerdere uitsneden werden onthouden). Voor optie B leidt het geven van meerdere eerder opgevraagde uitsneden tot meer benodigde verwerkingstijd van de bevraging aan de serverkant. De database bevraging wordt door elke toegevoegde uitsnede iets complexer, maar het aantal geselecteerde primitieven wordt kleiner. Het blijkt dat bij slechts een uitsnede onthouden de meeste winst wordt gehaald (vergelijk opties $B_{1\text{-}3}$ in figuur 4(a) en (b)) en dit is dus de beste oplossing voor optie B; dubbele overdracht wordt daarmee grotendeels voorkomen.

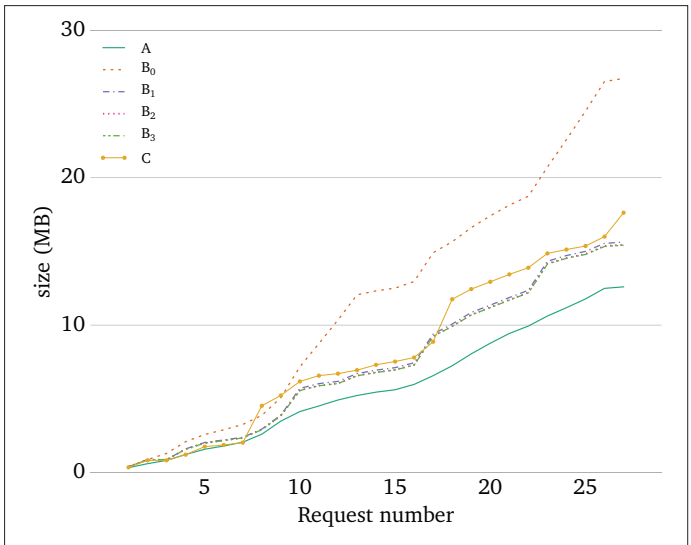
Optie C neemt de minste hoeveelheid tijd om over te sturen (en heeft ook het snelste moment van antwoord: time to first byte). Dit is gunstig, omdat hierna begonnen kan worden om de kaart incrementeel bij te werken. In verhouding met optie B_1 ligt de hoeveelheid datagebruik in dezelfde orde grootte (omdat data van de vorige uitsnede die getoond werd al aanwezig is op de client bij optie C ook niet nogmaals opgestuurd wordt). We hebben bij implementeren van optie C ook gemerkt dat het noodzakelijk is dat de data in goede



Figuur 3 - Gebruikersacties leiden tot een set van verzoeken, die afgevuurd worden p de vario-schaal server.



(a) Gemiddelde tijd voor ontvangst hele antwoord per verzoek (time to last byte).



(b) Hoeveelheid ontvangen data (cumulatief).

Figuur 4 - Verzamelde gegevens: (a) Tijd voor bevragen van de server en (b) de hoeveelheid overgedragen gegevens.

volgorde wordt afgehandeld, omdat anders de kaart niet in goede orde kan worden bijgewerkt. Dit is ook van toepassing als de gebruiker fanatiek de kaart gaat zoomen of schuiven. Het is belangrijk dat het systeem de vraag-antwoordverzoeken in de goede volgorde blijft afhandelen.

Al met al leveren de drie opties de mogelijkheid om vector data uit de tGAP structuur traploos te gebruiken. De meest geavanceerde optie (optie C) maakt het ook mogelijk om het kaartbeeld zeer geleidelijk object voor object bij te werken. Dit kan met gebruik van ongeveer dezelfde hoeveelheid bandbreedte en zelfde hoeveelheid verwerkingstijd als de rechttoe rechtaan optie A (geschikt voor niet tGAP-bewuste clients) gebruikt, maar levert een meer geleidelijke indruk, met snellere visuele feedback voor de eindgebruiker (door de tGAP-bewuste client).

Conclusie

We hebben drie alternatieve client-server communicatieopties ontworpen en geïmplementeerd. Onze experimenten laten zien dat het daarmee mogelijk is om data uit een varioschaal structuur op verschillende manieren via het web te ontsluiten.

Alle vraag-antwoord opties die onderzocht zijn, leveren een antwoord binnen een of twee seconde, en maken dus interactief kaartgebruik mogelijk. De inhoud van de kaart is hierbij traploos instelbaar qua hoeveelheid detail. De eindgebruiker wordt de mogelijkheid geboden om te beïnvloeden hoe druk het kaartbeeld is (en dus hoeveel data opgehaald en gevisualiseerd wordt). Dit kan zodoende worden gebaseerd op de directe voorkeur van de gebruiker. Mogelijk kan dit ook dynamisch worden bepaald door het systeem zelf, bijvoorbeeld als bandbreedte beperkt is om zo over te schakelen naar een minder gedetailleerde versie. Hiervoor zal het wel nodig zijn om statistieken bij te houden met betrekking tot de snelheid van de gegevensoverdracht.

Verder lieten de experimenten zien dat de mogelijkheid om dataverkeer te besparen door gebruik te maken van een cache al de meeste winst haalt door alleen hergebruik van het laatst ontvangen antwoord van de client, hetgeen zichtbaar is bij optie B_1. Een benadering die we verder willen onderzoeken, is het gebruik van 'eenvoudige' datablokken om beter onthouden van eerdere antwoorden mogelijk te maken. Eerste stappen in deze richting zijn inmiddels gezet met het afstudeerwerk van Adrie Rovers (Rovers, 2016).

Een andere richting die we graag verder willen onderzoeken, is de kaartinhoud nog meer geleidelijk te laten veranderen. Niet een heel vlakobject vervangen, maar ook bijvoorbeeld (delen van) de grenzen. Dit kan worden bereikt door gebruik te maken van expliciete 3D-geometrie, waarbij de z-coördinaat gebruikt wordt om op te slaan hoe de geometrie door verschillende schalen heen verandert. Met een redelijk standaard aanpak kan deze 3D ruimte-schaal geometrie efficiënt op de client worden afgebeeld met behulp van de grafische hardware (GPU). Eerste stappen in deze richting zijn gezet met het afstudeerwerk van Mattijs Driel (Driel, 2015).

Bronnen

- OGC, OpenGIS Web Map Server Interface Implementation Specification. URL portal.opengeospatial.org/files/?artifact_id=7196, 2000.
- Peter van Oosterom, Martijn Meijers, Variabele-schaal geo-informatie, Geo-Info, 9(10), pp. 14-19, 2012
- Adrie Rovers, Exploring the Use of a Generic Spatial Access Method for Caching and Efficient Retrieval of Vario-scale Data in a Client-Server Architecture, Master's thesis, Delft University of Technology, pp. 101, 2016.
- Mattijs Driel, Real time intersections on space scale cube data, Master's thesis, Utrecht University, pp. 12, 2015.

Dit artikel is een bewerking van het Engelstalige artikel: Lina Huang, Martijn Meijers, Radan Šuba, Peter van Oosterom, Engineering web maps with gradual content zoom based on streaming vector data, In: ISPRS Journal of Photogrammetry and Remote Sensing, Elsevier BV, 114, pp. 274-293, 2016.

Radan Šuba is promovendus GIS technologie bij de TU Delft. Hij is bereikbaar via R.Suba@tudelft.nl.

Martijn Meijers is onderzoeker GIS technologie bij de TU Delft. Hij is bereikbaar via B.M.Meijers@tudelft.nl.

Peter van Oosterom is professor GIS technologie bij de TU Delft. Hij is bereikbaar via P.J.M.vanOosterom@tudelft.nl.

Lina Huang is docente bij de Wuhan Universiteit (en was in 2014 gastonderzoekster bij de TU Delft). Zij is bereikbaar via linahuang@whu.edu.cn.

Ook tatoeages van kaarten

In Geo-Info nummer 4, 2016 is een pagina gewijd aan tatoeages met vakinstrumenten. Beloofd was opname van een foto van de eerste vrouwelijke tatoeagedrager van een landmeetkundig instrument op GeoBuzz, maar die werd (nog?) niet gesignaleerd. In reacties moest verder toegegeven worden dat er nog veel meer tatoeages zijn met globes en landkaarten (meestal kleinschalige). Volstaan wordt hier met enkele voorbeelden van www.popsugar.com. De ene is een hoogtelijnenkaart van de Mount St. Helens en de andere is er een met midden- tot grootschalige topografie.

Adri den Boer

Landmeters en geodeten...

Een kruiswoordpuzzel in NRC Handelsblad van 29 september 2016 (pag. C19) leerde: Verticaal 3: heet officieel 'geodeet'. Antwoord: landmeter.

Soms is er toch verschil. In Geo-Info 2017-1 zijn in de bijdrage 'Vakidiotie en T-shirts' uitsluitend kledingstukken van surveyors, landmeters, opgenomen. Maar er zijn er ook van geodeten (www.cafepress.co.uk)...

Adri den Boer